ธาวินทร์ ลีลาคุณารักษ์

SOFT MAIL

Cascade-DB อีกทางเลือกหนึ่งของระบบบริหารฐานข้อมูลขนาดเล็ก

Cascade-DB โปรแกรมบริหารฐานข้อมูลที่ออกแบบมาสำหรับใช้กับระบบงานขนาดเล็ก

ปกติเวลาเราเขียนโปรแกรมที่มีการเรียกใช้ตารางข้อมูลหรือฐานข้อมูลบนวินโดวส์ การทำงานที่เกี่ยวกับการจัดการข้อมูลในตารางข้อมูลของโปรแกรมที่เราสร้างขึ้นนั้น ส่วนใหญ่จะดำเนินการผ่านกลไกสำหรับจัดการฐานข้อมูล (Database Engine) ที่เชื่อมต่อไว้กับ ODBC ของวินโดวส์หรือ Borland Database Engine อย่างใดอย่างหนึ่ง ซึ่งตารางข้อมูลที่เรารู้จักและคุ้นเคยกันดี ได้แก่ ตารางข้อมูลแบบ Access (.DB) ตารางข้อมูลแบบ Paradox (.DB) หรือตารางข้อมูลแบบ dBASE/Foxpro (.DBF) เป็นต้น

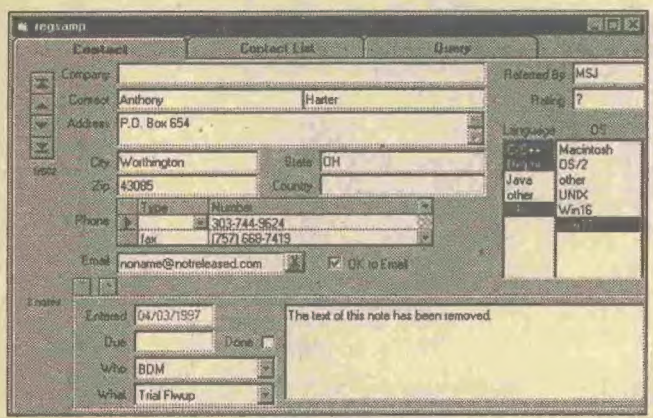
โครงสร้างการจัดการกับข้อมูลที่อยู่ในตารางข้อมูลเหล่านี้ จะนำแนวคิดของการบริหารฐานข้อมูลแบบ Relational Database หรือที่เรียกกันทั่วไปว่า เป็นฐานข้อมูลแบบ SQL มาใช้ แต่เนื่องจากข้อจำกัดของเครื่องคอมพิวเตอร์แบบพีซีในยุคก่อน ๆ ทำให้การออกแบบระบบจัดการฐานข้อมูลของตารางข้อมูลแบบต่าง ๆ ที่กล่าวมาในข้างต้น ไม่สามารถใช้แนวคิดของฐานข้อมูลแบบ Relational Database ได้อย่างสมบูรณ์ ซึ่งถ้าเรานำไปเปรียบเทียบกับระบบจัดการฐานข้อมูลแบบ SQL ระดับ Hight-End อย่าง Oracle หรือ MS-SQL เราจะเห็นว่า ประสิทธิภาพและความสามารถในด้านต่าง ๆ ของระบบจัดการฐานข้อมูลระดับล่างไม่สามารถเทียบกับระบบบริหารฐานข้อมูลระดับสูงดังกล่าวได้เลย

Cascade-DB เป็นโปรแกรมบริหารฐานข้อมูล (DBMS) ใช้แนวคิดของ Ralational Database หรือฐานข้อมูลแบบ SQL เช่นเดียวกัน แต่เป็นระบบที่ออกแบบมาสำหรับใช้กับระบบงานขนาดเล็ก ที่มีแนวคิดการจัดการฐานข้อมูลที่ดีกว่าระบบจัดการฐานข้อมูลระดับล่างอย่างฐานข้อมูลแบบ Access ฐานข้อมูลแบบ Paradox หรือฐานข้อมูลแบบ dBASE/Foxpro ที่เราใช้กันอยู่ โดยในการจัดการกับข้อมูลที่อยู่ภายในฐานข้อมูลของ Cascade-DB จะเป็นคำสั่ง SQL ทั้งหมด

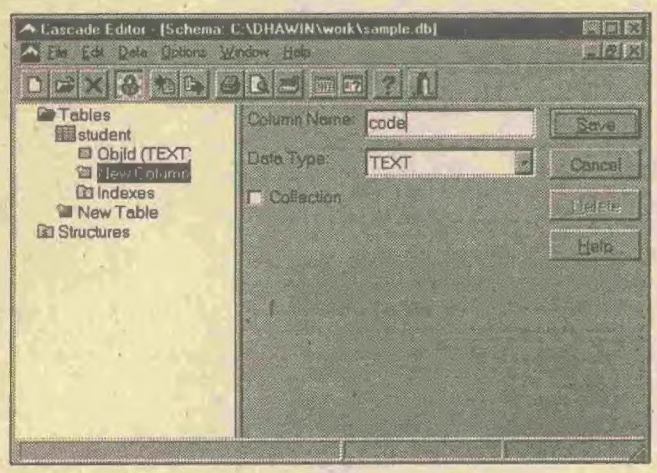
เมื่อเราติดตั้ง Cascade-DB ลงในเครื่องคอมพิวเตอร์ จะพบว่าในชุดโปรแกรมของ Cascade-DB จะมีเพียงกลไกสำหรับจัดการฐานข้อมูล (Database Engine) ของ Cascade-DB ที่ใช้เชื่อมต่อกับ ODBC ของวินโดวส์หรือ Borland Database Engine และโปรแกรม Cascade Editor มาให้เท่านั้น ส่วนเครื่องมืออื่น ๆ เช่น เครื่องมือสำหรับสร้างส่วนยูสเซอร์อินเทอร์เฟสหรือเอดิเตอร์สำหรับสร้างโปรแกรม จะไม่มีอยู่ในโปรแกรมชุดนี้เลย

ทั้งนี้เนื่องจากทางผู้ผลิตโปรแกรม Cascade-DB ต้องการให้ระบบจัดการฐานข้อมูลของตนสามารถใช้กับภาษาโปรแกรมภาษาใดก็ได้ ซึ่งขึ้นอยู่กับความถนัดของโปรแกรมเมอร์ ในไดเรคทอรีของ Cascade-DB จะมีไลบรารีและเครื่องมือที่จำเป็นในการจัดการฐานข้อมูลสำหรับคอมพายเลอร์ที่นิยมใช้บนวินโดวส์หลายรูปแบบด้วยกัน ดังนั้นไม่ว่าเราจะถนัด Visual Basic, Delphi หรือ C++ ก็สามารถสร้างโปรแกรมเรียกใช้ฐานข้อมูลของ Cascade-DB ได้เช่นเดียวกัน โดย Cascade-DB ได้เตรียมคำแนะนำการใช้ฐานข้อมูลของ Cascade-DB ร่วมกับเครื่องมือพัฒนาโปรแกรมแต่ละแบบไว้อย่างชัดเจน

Cascade-DB อีกทางเลือกหนึ่ง
ของระบบบริหารฐานข้อมูลขนาดเล็ก



รูปที่ 1 ตัวอย่างแอพพลิเคชันที่ใช้ฐานข้อมูลของ Cascade-DB ที่สร้างด้วย Visual Basic

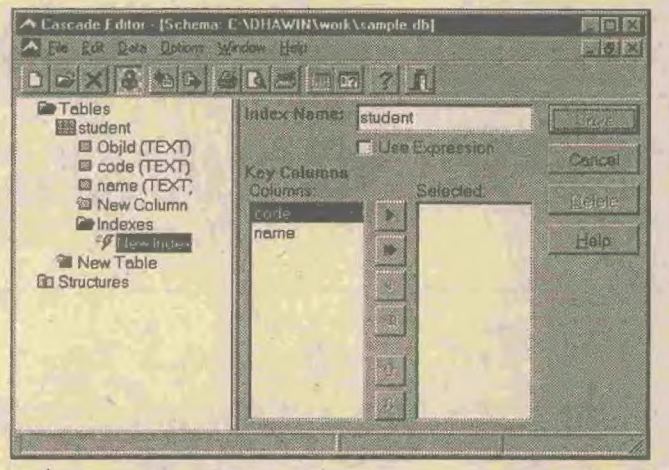


รูปที่ 2 การกำหนดฟิลด์ให้กับตารางข้อมูล

ทดลองสร้างโปรแกรม

สำหรับตัวอย่างที่ผมนำมาเสนอประกอบบทความชุดนี้ จะเป็นโปรแกรมที่มีการเรียกใช้ฐานข้อมูลของ Cascade-DB โดยใช้ Delphi 2.0 ในการจัดทำ เนื้อหาของโปรแกรมตัวอย่างชุดนี้จะเป็นโปรแกรมที่ใช้สำหรับบันทึกชื่อที่อยู่เบอร์โทรศัพท์ของเพื่อน ๆ โดยฟิลด์หรือคอลัมน์ของตารางข้อมูล (Myfriend) ที่เราจะใช้กับโปรแกรม จะเป็นข้อมูลแบบ Text ทั้งหมด ซึ่งมีรายละเอียดดังนี้

- ฟิลด์ name ใช้สำหรับเก็บชื่อ
- ฟิลด์ lastname ใช้สำหรับเก็บนามสกุล
- ฟิลด์ address ใช้สำหรับเก็บบ้านเลขที่และหมู่บ้าน
- ฟิลด์ tambol ใช้สำหรับเก็บชื่อตำบลหรือชื่อแขวง
- ฟิลด์ amphor ใช้สำหรับเก็บชื่ออำเภอหรือชื่อเขต
- ฟิลด์ changwat ใช้สำหรับเก็บชื่อจังหวัด
- ฟิลด์ postcode ใช้สำหรับเก็บรหัสไปรษณีย์
- ฟิลด์ phone ใช้สำหรับเก็บเบอร์โทรศัพท์



รูปที่ 3 การสร้างอินเด็กซ์ให้กับตารางข้อมูล

การสร้างฐานข้อมูลและตารางข้อมูล

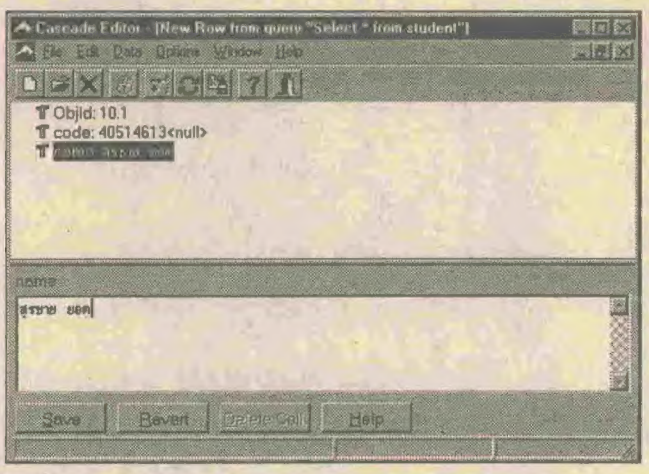
ลักษณะฐานข้อมูลของ Cascade-DB จะต่างจากฐานข้อมูลแบบ Access, Paradox หรือ dBASE/Foxpro ตรงที่ตัวแฟ้มสกุล .db ของ Cascade-DB จะเป็นตัวฐานข้อมูลทั้งชุด ซึ่งภายในฐานข้อมูลจะประกอบด้วยตารางข้อมูล (Table) อย่างน้อยหนึ่งตาราง แต่แฟ้ม .db ของ Access หรือ Paradox รวมถึงแฟ้ม .dbf ของ dBASE/Foxpro จะเป็นที่เก็บตารางข้อมูลเพียงหนึ่งตารางเท่านั้น

ในรูปที่ 2 และรูปที่ 3 เป็นตัวอย่างการสร้างองค์ประกอบต่าง ๆ ของฐานข้อมูลของ Cascade-DB ซึ่งงานทั้งหมดจะทำบนเครื่องมือที่เรียกว่า Cascade Editor ที่ Cascade-DB เตรียมไว้ให้ โดยในขั้นตอนแรกของการสร้างฐานข้อมูลชุดใหม่ โปรแกรม Cascade Editor จะให้เราใส่ชื่อของฐานข้อมูลก่อน ในตัวอย่างชุดนี้จะจัดเก็บในแฟ้มชื่อ SAMPLE.DB

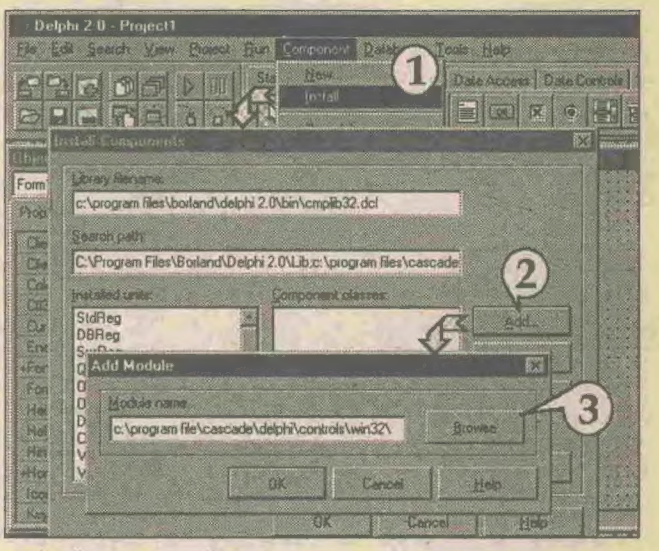
สำหรับขั้นตอนถัดมาจะเป็นการสร้างตารางข้อมูล (Table) ที่เก็บอยู่ในฐานข้อมูล ฟิลด์ของตารางข้อมูล และ Indexes ให้กับตารางข้อมูลแต่ละชุดตามลำดับ โดยโปรแกรม Cascade Editor จะแสดงองค์ประกอบของสิ่งที่เราจะจัดทำในรูปของโครงร่างแขนง (Tree) เป็นลำดับชั้นลงมา ซึ่งในการสร้างส่วนต่าง ๆ ของตารางข้อมูลที่กล่าวมาในข้างต้น จะมีขั้นตอนไม่ยุ่งยากครับ

สมมติว่าเราต้องการจะสร้างฟิลด์ข้อมูลฟิลด์ใหม่ให้กับตารางข้อมูลชุดหนึ่ง ขั้นตอนจะเริ่มจากใช้เม้าส์ขยายโครงร่างแขนงของชื่อตารางข้อมูลที่เราจะสร้างฟิลด์ให้ขยายออก (ดับเบิ้ลคลิกที่ชื่อของตาราง) แล้วใช้เม้าส์คลิกที่หัวข้อ "New Column" จากนั้นจะมาใส่ชื่อฟิลด์ในช่อง "Column Name" และเลือกชนิดของข้อมูลในช่อง Data Type ของโปรแกรมดังตัวอย่างในรูปที่ 2 แล้วมาคลิกที่ปุ่มคำสั่ง Save ตามลำดับ สำหรับงานอื่น ๆ เช่น การสร้างตารางข้อมูลชุดใหม่ หรือการสร้าง Indexes ก็จะดำเนินการในลักษณะเดียวกันนี้

Cascade-DB อีกทางเลือกหนึ่ง
ของระบบบริหารฐานข้อมูลขนาดเล็ก

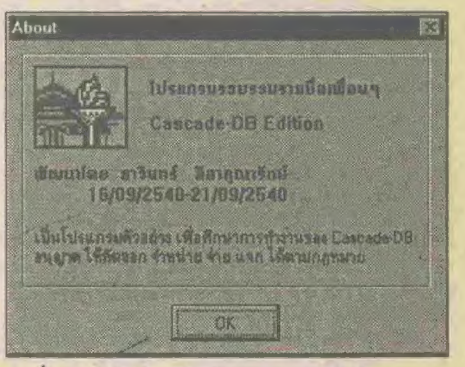


รูปที่ 4 การบันทึกข้อมูลลงในตารางข้อมูลบน Cascade Editor



รูปที่ 5 ขั้นตอนการติดตั้ง Component ของ Cascade-DB บน Delphi

รูปที่ 6 หน้าจอหลักของโปรแกรมประกอบบทความ



รูปที่ 7 หน้าจอย่อยของโปรแกรมตัวอย่าง

การใส่ข้อมูลลงในตารางข้อมูล

บนโปรแกรม Cascade Editor ของ Cascade-DB นี้ เราสามารถจะวิวดูข้อมูลจากตารางข้อมูลและสามารถใช้โปรแกรม Cascade Editor บันทึกข้อมูลใส่ลงในตารางข้อมูลได้ด้วย โดยในการวิวดูข้อมูลในตารางข้อมูล เราจะใช้เม้าส์คลิกที่ชื่อตารางข้อมูลที่เราต้องการ แล้วใช้เม้าส์คลิกที่ปุ่มคำสั่ง [view.bmp] ตามลำดับ ส่วนขั้นตอนการบันทึกข้อมูลลงในตารางข้อมูลนั้น เราจะต้องวิวตารางข้อมูลที่เราจะใส่ข้อมูลขึ้นมาก่อน แล้วใช้เม้าส์คลิกที่ปุ่มคำสั่ง Add Row ที่ปรากฏอยู่บนจอตามลำดับ โปรแกรม Cascade Editor จะแสดงหน้าจอสำหรับบันทึกข้อมูลลงในตารางข้อมูลดังตัวอย่างในรูปที่ 4 โดยในการใส่ข้อมูลเราจะใช้เม้าส์คลิกที่ชื่อคอลัมน์ที่เราต้องการ จากนั้นจะมาใส่ข้อมูลลงในกรอบที่อยู่ด้านล่างของหน้าจอจนกว่าจะครบทุกคอลัมน์

Cascade-DB กับ Delphi

ก่อนที่เราจะดำเนินการสร้างโปรแกรมที่มีการเรียกใช้ฐานข้อมูลของ Cascade-DB เราจำเป็นต้องจัดองค์ประกอบของคอมพายเลอร์ที่เราใช้ให้รู้จักกับกลไกของ Cascade-DB ก่อน โดยการติดตั้งไลบรารีและ Control Pallate ที่ Cascade-DB เตรียมไว้ให้ โดยดูขั้นตอนจากคำแนะนำที่มากับโปรแกรม

สำหรับการติดตั้งองค์ประกอบของ Cascade-DB เพื่อใช้งานบน Delphi จะมีการติดตั้ง Component Pallate ของ Cascade-DB เพียงอย่างเดียว โดยขั้นตอนในการติดตั้งจะเริ่มจาก เลือกหัวข้อ "Component" จากเมนูหลักของ Delphi แล้วเลือกหัวข้อ "Install" จากเมนูที่ปรากฏขึ้นตามลำดับ จากนั้นจะใช้เม้าส์คลิกที่ปุ่มคำสั่ง Add ของไดอะล็อก Install Component ที่ปรากฏอยู่บนจอ Delphi จะแสดงไดอะล็อก Add Module ซ้อนขึ้นมาอีกชุดหนึ่ง ให้ Browse ไปที่ไดเรคทอรี C:file.pas แล้วคลิกที่ปุ่มคำสั่ง OK ของไดอะล็อกทั้งสองชุดตามลำดับ หลังจากนั้น Delphi จะทำการคอมพายล์งาน สร้างไลบรารี พร้อมกับติดตั้งชุดต้นแบบชิ้นส่วนโปรแกรม (Component Pattate) ให้ (ดูรูปที่ 5)

สร้างหน้าจอของตัวอย่าง

ในตัวอย่างชุดนี้จะประกอบด้วยหน้าจอของโปรแกรมจำนวน 2 ชุดด้วยกัน โดยกิจกรรมต่าง ๆ ที่เกี่ยวข้องกับการใช้ฐานข้อมูลของ Cascade-DB จะอยู่ที่หน้าจอชุดแรกทั้งหมด (รูปที่ 6) ส่วนหน้าจออีกชุดหนึ่ง (รูปที่ 7) จะเป็นหน้าจอที่ให้ข่าวสารเล็ก ๆ น้อย ๆ ไม่สำคัญอะไรนัก โดยหน้าจอชุดหลังนี้จะเกิดจากการสร้างหน้าจอแบบ About ของ Delphi เราเพียงแต่เปลี่ยนข้อความในจุดต่าง ๆ ให้ตรงกับที่เราต้องการเท่านั้น สำหรับหน้าจอชุดที่สองนี้ ผมจะขอข้ามรายละเอียดการจัดองค์ประกอบของหน้าจอไป ส่วนจุดที่มีการแทรกลงในแฟ้ม .PAS ของหน้าจอชุดนี้ จะแสดงอยู่ในลิสติงที่ 2 โดยในจุดที่มีการแทรกชุดคำสั่ง ผมจะแสดงเป็นตัวอักษรแบบตัวหนาไว้ เพื่อให้สังเกตได้ง่ายขึ้น

สำหรับหน้าจอชุดแรกจะเป็นหน้าจอแบบไดอะล็อก และเวลาที่หน้าจอปรากฏบนจอภาพ จะอยู่ตรงกลางจอพอดี ซึ่งได้จากการเปลี่ยนค่าคุณสมบัติ (Properties) ของ Form หัวข้อ "BorderStyle" จากเดิมเป็น "bsDialog" และเปลี่ยนค่าคุณสมบัติหัวข้อ "Position" จากเดิมเป็น "psScreenCenter" ตามลำดับ

ส่วนข้อความที่ปรากฏบนแถบไตเติลของฟอร์ม (คุณสมบัติหัวข้อ Caption) จะเปลี่ยนเป็น "โปรแกรมรวบรวมรายชื่อเพื่อน ๆ" สำหรับชิ้นส่วนของโปรแกรมแบบอื่น ๆ ที่ปรากฏบนหน้าจอชุดนี้จะประกอบด้วย ชิ้นส่วนโปรแกรมแบบ Panel จะใช้สำหรับแสดงแถบชื่อของโปรแกรม ชิ้นส่วนโปรแกรมแบบ Label ใช้แสดงคำอธิบายฟิลด์ข้อมูล และชิ้นส่วนโปรแกรมแบบ Button สำหรับสร้างเป็นปุ่มคำสั่ง "About" และปุ่มคำสั่ง "เลิกงาน" ซึ่งชิ้นส่วนของโปรแกรมที่กล่าวมาในข้างต้นนี้ จะได้จากต้นแบบชิ้นส่วนโปรแกรมชุด Standard ทั้งหมด โดยรายละเอียดการปรับแต่งคุณสมบัติของชิ้นส่วนโปรแกรมแต่ละชิ้นส่วน จะอยู่ในตารางที่ 1

ตารางที่ 1

ชิ้นส่วนโปรแกรม	คุณสมบัติ (Properties)	เปลี่ยนค่าจากเดิมเป็น
Panel(แถบชื่อโปรแกรม)	BevelInner	bvLowered'
Label(ชื่อ)	Caption	โปรแกรมรวมรายชื่อเพื่อนๆ
Label(นามสกุล)	Color	Cascade-DB Edition'
Label(ที่อยู่)	Font	clYellow
Label(ตำบล)	Caption	EucrosiaUPC สีน้ำเงิน ตัวหนา
Label(อำเภอ)	Caption	ขนาด24 'ชื่อ :'
Label(จังหวัด)	Caption	'นามสกุล :'
Label(โทร)	Caption	'ที่อยู่ :'
Button(เลิกงาน)	Caption	'ตำบล :' 'อำเภอ:' 'จังหวัด :'
Button(About)	Caption	'โทร :'
	Caption	'เลิกงาน'
	Caption	'About'
	Caption	

ส่วนชิ้นส่วนของโปรแกรมอีกกลุ่มหนึ่ง จะเป็นชิ้นส่วนของโปรแกรมที่ได้จากต้นแบบชิ้นส่วนชุด Cascade ที่เราติดตั้งขึ้นมาใหม่ โดยจะประกอบด้วยชิ้นส่วนโปรแกรมแบบ CdbData สำหรับควบคุมการจัดการกับข้อมูลในฐานข้อมูลของ Cascade-DB ชิ้นส่วนโปรแกรมแบบ CdbEdit สำหรับรับค่าไปเก็บในฟิลด์ต่าง ๆ ของตารางข้อมูล และชิ้นส่วนโปรแกรมแบบ CdbGrid สำหรับแสดงข้อมูลที่อยู่ในตารางข้อมูลของ Cascade-DB ในรูปของตารางบนจอภาพ โดยเราจะเลือกให้ปรากฏบนจอเฉพาะฟิลด์ "name", "lastname" และ "phone" เท่านั้น ส่วนรายละเอียดการปรับแต่งคุณสมบัติ (Properties) ของชิ้นส่วนของโปรแกรมกลุ่มหลังนี้จะอยู่ในตารางที่ 2

ตารางที่ 2

ชิ้นส่วนโปรแกรม	คุณสมบัติ (Properties)	เปลี่ยนค่าจากเดิมเป็น
CdbData1	DatabaseName	'C:.db' SELECT *
CdbEdit(ชื่อ)	RecordSource	FROM myfriend ORDER BY
CdbEdit(นามสกุล)	DataControl	ObjIdCdbData1
CdbEdit(ที่อยู่)	DataFieldName	'name'
CdbEdit(ตำบล)	DataControl	CdbData1
CdbEdit(อำเภอ)	DataFieldName	'lastname'
CdbEdit(จังหวัด)	DataControl	CdbData1
CdbEdit( )	DataFieldName	'address'
CdbEdit(โทร)	DataControl	CdbData1
CdbGrid1	DataFieldName	'tambol'
	DataControl	CdbData1
	DataFieldName	'amphor'
	DataControl	CdbData1
	DataFieldName	'chagwat'
	DataControl	CdbData1
	DataFieldName	'postcode'
	DataControl	CdbData1
	DataFieldName	'phone'
	DataControl	CdbData1
	CDBColumns	'name','lastname','phone'

Cascade-DB อีกทางเลือกหนึ่ง
ของระบบบริหารฐานข้อมูลขนาดเล็ก

การแทรกชุดคำสั่งควบคุมโปรแกรม

โปรแกรมชุดนี้ จะมีจุดที่เราต้องแทรกชุดคำสั่งควบคุมการทำงานของโปรแกรมเพียง 2 จุด คือ ที่ปุ่มคำสั่ง "About" และที่ปุ่มคำสั่ง "เลิกงาน" โดยที่ปุ่มคำสั่ง About จะใช้สำหรับเรียกหน้าจอชุดที่สองขึ้นมาทำงาน ชุดคำสั่งที่จะแทรกลงไป (โดยดับเบิ้ลคลิกที่ภาพปุ่มคำสั่ง) คือ คำสั่ง About; (บรรทัดที่ 054 ของลิสติงที่ 1) และใส่ชื่อยูนิตของหน้าจอชุดที่สองไว้ที่ส่วน Uses ของหน้าจอชุดแรก (บรรทัดที่ 008 ของลิสติงที่ 1 จุดที่เป็นตัวหนา) ส่วนที่ปุ่มคำสั่ง "เลิกงาน" จะเป็นปุ่มคำสั่งสำหรับปิดโปรแกรม ชุดคำสั่งที่เราจะแทรกลงไปก็คือคำสั่ง Close; (บรรทัดที่ 049 ของลิสติงที่ 1) ส่วนชุดคำสั่งที่เกี่ยวกับการจัดการฐานข้อมูลของ Cascade-DB ไม่ได้แสดงไว้ในตัวอย่างนี้ เนื่องจากองค์ประกอบต่าง ๆ ของโปรแกรมที่เราจัดไว้ มีกลไกภายในที่ช่วยให้โปรแกรมสามารถทำงานได้อย่างสมบูรณ์อยู่แล้ว ไม่ว่าจะเป็นการเพิ่มรายการในตารางข้อมูล การแก้ไขรายการ การลบรายการ หรือการเลื่อนตัวชี้รายการไปยังจุดต่าง ๆ ของตาราง ซึ่งงานทั้งหมดนี้จะอยู่ในชิ้นส่วนโปรแกรม CdbData ทั้งหมด ก่อนจะจบผมต้องขออภัยที่ผมไม่มีตัวอย่างโปรแกรมที่ใช้ฐานข้อมูลของ Cascade-DB ที่สร้างด้วย Visual Basic หรือ C++ เนื่องจากผมไม่มีคอมพายเลอร์ทั้งสองชุดนี้อยู่ในมือ แต่ก็เชื่อว่าตัวอย่างที่ผมนำมาเสนอนี้ คงจะพอช่วยให้ท่านผู้อ่านเห็นแนวทางการนำ Cascade-DB ไปใช้งานบ้างตามสมควร

การสร้างโปรแกรมที่มีการเรียกใช้ฐานข้อมูลของ Cascade-DB จะมีเทคนิคเล็ก ๆ น้อย ๆ ที่ช่วยให้การสร้างโปรแกรมทำได้ง่ายขึ้นคือ เราควรจะใส่ข้อมูลตัวอย่างลงในตารางข้อมูลที่อยู่ในฐานข้อมูลของ Cascade-DB ทุกตาราง อย่างน้อยตารางละหนึ่งรายการ ส่วนการนำโปรแกรมประกอบบทความชุดนี้ไปทดลองใช้ ต้องแน่ใจก่อนนะครับว่าในเครื่องของท่านมี engine จัดการฐานข้อมูลของ Cascade-DB อยู่ในเครื่องแล้ว ถ้าไม่มีโปรแกรมจะทำงานไม่ได้ครับ

ลิสติงที่ 1
SCREEN.PAS

```
001: unit screen;
002:
003: interface
004:
005: uses
006: Windows, Messages, SysUtils, Classes, Graphics, Controls,
007: Forms, Dialogs, Grids, CdbGrid, CdbData, StdCtrls, ExtCtrls,
008: Mask, CdbEdit, xabout; เรียกยูนิตหน้าจอ about
009:
010: type
011: TForm1 = class(TForm)
012: Panel1: TPanel; แถบชื่อโปรแกรม
013: Button4: TButton; ปุ่มคำสั่ง "เลิกงาน"
014: Label1: TLabel; "ชื่อ"
015: Label2: TLabel; "นามสกุล"
016: Label3: TLabel; "ที่อยู่"
017: Label4: TLabel; "ตำบล"
018: Label5: TLabel; "อำเภอ"
019: Label6: TLabel; "จังหวัด"
020: Label7: TLabel; "โทร"
021: CdbData1: TCdbData; ปุ่มควบคุมการจัดการฐานข้อมูล
022: CdbEdit1: TCdbEdit; รับข้อมูลในช่อง ชื่อ
023: CdbEdit2: TCdbEdit; รับข้อมูลในช่อง นามสกุล
024: CdbEdit3: TCdbEdit; รับข้อมูลในช่อง ที่อยู่
025: CdbEdit4: TCdbEdit; รับข้อมูลในช่อง ตำบล
026: CdbEdit5: TCdbEdit; รับข้อมูลในช่อง อำเภอ
027: CdbEdit6: TCdbEdit; รับข้อมูลในช่อง จังหวัด
028: CdbEdit7: TCdbEdit; รับข้อมูลในช่อง รหัสไปรษณีย์(ไม่มีข้อความ)
029: CdbGrid1: TCdbGrid; แสดงรูปตารางข้อมูล
030: CdbEdit8: TCdbEdit; รับข้อมูลในช่อง โทร
031: Button1: TButton; ปุ่มคำสั่ง "About"
032: procedure Button4Click(Sender: TObject);
033: procedure Button1Click(Sender: TObject);
034: private
035: Private declarations
036: public
037: Public declarations
038: end;
039:
040: var
041: Form1: TForm1;
042:
043: implementation
044:
045: $R *.DFM
```

→

ลิสติงที่ 1 (ต่อ)

```
046:
047: procedure TForm1.Button4Click(Sender: TObject);
048: begin
049: close;
050: end;
051:
052: procedure TForm1.Button1Click(Sender: TObject);
053: begin
054: about;
055: end;
056:
057: end.
```

ลิสติงที่ 2
XABOUT.PAS

```
001: unit xabout;
002:
003: interface
004:
005: uses Windows, SysUtils, Classes, Graphics, Forms, Controls, StdCtrls,
006: Buttons, ExtCtrls;
007:
008: type
009: TAboutBox = class(TForm)
010: Panel1: TPanel;
011: ProgramIcon: TImage;
012: ProductName: TLabel;
013: Version: TLabel;
014: Copyright: TLabel;
015: Comments: TLabel;
016: OKButton: TButton;
017: procedure OKButtonClick(Sender: TObject);
018: private
019: Private declarations
020: public
021: Public declarations
022: end;
023:
024: var
025: AboutBox: TAboutBox;
026:
027: procedure about;
028:
029: implementation
030:
031: $R *.DFM
032:
033: procedure about;
034: begin
035: AboutBox.visible := True;
036: end;
037:
038: procedure TAboutBox.OKButtonClick(Sender: TObject);
039: begin
040: close;
041: end;
042:
043: end.
```